From: **Waldemar Horwat** waldemar@google.com
Subject: Substatement function definitions
Date: March 21, 2008 at 2:19 PM
To: Maciej Stachowiak mjs@apple.com
Cc: Brendan Eich brendan@mozilla.org, Lars Hansen lhansen@adobe.com, Mark S. Miller erights@google.com, Pratap Lakshman (VJ#SDK) pratapl@microsoft.com, crock@yahoo-inc.com, Kris Zyp kzyp@sitepen.com, Mike Cowlishaw MFC@uk.ibm.com, Allen Wirfs-Brock Allen.Wirfs-Brock@microsoft.com, Adam Peller apeller@us.ibm.com, ggaren@apple.com, e-TC39@ecma-international.org

Maciej Stachowiak wrote:

> If function declarations in statement position were somehow harmful or ill-conceived, then maybe you would have a point. But they are not. There is nothing wrong with the idea conceptually, and were they to be standardized, no one would be hanging their head in shame years later.

I'm responsible for function declarations not being allowed in inner blocks or substatements within ES3. These were disallowed by the standard because they turned out to be conceptually harmful and fixing the problems was more than the committee wanted to take on back then.

There were three points:

1. Hoisting such declarations to the top level (as is done with var) doesn't work because such functions can capture scopes that include variables that don't exist yet; ES3 didn't have local scopes but it did have exception scopes which cause the same problem. It got worse when we considered what would happen once we extended the language to have constants and dynamic (i.e. run-time) type annotations -- such functions could capture uncreated constants and, worse, variables whose types hadn't been computed yet!

2. Binding such declarations only as they're encountered would have worked fine but we didn't want to implement this local binding in ES3 just for support of functions.

3. If such declarations were in the substatement position of an if statement, the planned intent was to create them only if the if expression was true (or false for an else clause), and put them into the nearest enclosing block scope. This would constitute a form of conditional compilation. A block with an attribute before it would be a non-scoping block that distributes the attribute to the contained definitions, so you could attach several definitions to one if statement.

Should we choose to implement these in ES3.1, they should be locally bound (as in point 2) and not hoisted. We can continue to disallow the behavior in point 3 unless we want to take on the work to support such conditional definitions in ES3.1.

Waldemar